Undecimated wavelet transforms for image de-noising

Aglika Gyaourova
Department of Computer Science, University of Nevada, Reno
aglika@cs.unr.edu

Chandrika Kamath and Imola K. Fodor
Center for Applied Scientific Computing, Lawrence Livermore National Laboratory
P.O. Box 808, L-560, Livermore, CA 94551
kamath2@llnl.gov and fodor1@llnl.gov

November 19, 2002

Abstract

A few different approaches exist for computing undecimated wavelet transform. In this work we construct three undecimated schemes and evaluate their performance for image noise reduction. We use standard wavelet based de-noising techniques and compare the performance of our algorithms with the original undecimated wavelet transform, as well as with the decimated wavelet transform. The experiments we have made show that our algorithms have better noise removal/blurring ratio.

1 Introduction

Almost every kind of data contains noise. Noise reduction is a required step for any sophisticated algorithms in computer vision and image processing. This problem has existed for a long time and yet there is no good enough solution for it. A tradeoff between the removed noise and the blurring in the image always exist. The use of wavelet transform for signal de-noising has been started in last decade. Wavelets capability to give detail spatial-frequency information is the main reason for this investigation. This property promises a possibility for better discrimination between the noise and the real data. Successful exploitation of wavelet transform might lessen the blurring effect or even overcome it completely.

There are two main types of wavelet transform - continuous and discrete [2]. Because of computers discrete nature, computer programs use the discrete wavelet transform. The discrete transform is very efficient from the computational point of view. Its only drawback is that it is not translation invariant. Translations of the original signal lead to different wavelet coefficients. In order to overcome this and to get more complete characteristic of the analyzed signal the undecimated wavelet transform was proposed. The general idea behind it is that it doesn't decimate the signal. Thus it produces more precise information for the frequency localization. From the computational point of view the undecimated wavelet transform has larger storage space requirements and involves more computations.

Two main algorithms for computing the undecimated wavelet transform exist. The algorithm à trous [6] and Beylkin's algorithm [1]. These two algorithms approach the problem from different ways. Another undecimated algorithm exists that has been discovered in search for completely different properties [10]. For reviews on the topic see [5], [12], and [13].

This paper is organized as follows. Section 2 and Section 3 give brief descriptions of the à trous and Beylkin's algorithms, which are the classical undecimated algorithms. Each of the Sections 4, 5, and 6 explains one of the non-decimated algorithms that we have used in our experiments. Section 7 gives the outline of the wavelet based denoising algorithms. Section 8 describes the experimental results and shows some of the images and the measurements. Finally, Section 9 summarizes the results and the observations we have made while working on that project.

2 The à trous algorithm

The most often mentioned algorithm is the "algorithm à trous". It modifies the standard Discrete Wavelet Transform (DWT) decomposition scheme by modifying the low pass and high pass filters at each consecutive level. It imitates the sub-sampling of the filtered signal by up-sampling the low-pass filter. This up-sampling is done by including zeros between each of the filter's coefficients at each level. (The French word *trous* means *holes*.) The detail coefficients are computed as the difference between the low passed images from two consecutive levels. The inverse transform is computed by adding the detail coefficients from all levels to the final low-resolution image. We have not implemented and tested the performance of this algorithm.

3 Beylkin's algorithm

The non-decimation approach is one way to think about the undecimated wavelet transform. Another way is to think about the shift invariance. To be completely shift invariant the transform has to be computed for all possible shifts of the original signal. G. Beylkin [1] first realized that the coefficients from *all* shifts are not necessary. In fact, the shifts by any odd number will give the same coefficients as the shift by one. The same is true for any even number – it is equivalent to a shift by zero. Generally, this means that computing the DWT for the signal and its shift by one and repeating this for each level, will give **all** possible wavelet coefficients. (For two-dimensional signals the shift has to be extended to a shift by one in each direction - horizontal, vertical and diagonal.)

The inverse of this transform is not unique. Because of the shifts, we have redundant information. Instead of having N coefficients as in the DWT, with the undecimated transform their number is approximately $NlogN$ [1]. The original signal can be exactly reconstructed from any set of corresponding shifted coefficients. One approach for constructing the inverse transform is to use a rule to decide which coefficients to throw away and which to keep. The goal is to eliminate the redundancy and end up with N coefficients. Then we can perform the inverse of the DWT using the coefficients that are left. This technique is usually compared with the best basis algorithm for wavelet packets. Wavelet packets are another way to have redundant information in the wavelet coefficients, see [14], [5]. The difficulty with that approach is in defining the decision rule. Good decision rules are either very computationally expensive or totally disregarding the spatial relationships of the coefficients. Another inverse transform technique is to make the independent inverses for each shift and then to average the separate results into one [12], [9]. If the coefficients are not changed the reconstructed image will be exactly the same as the original. So we just have made number-of-shifts times more work. However, if the coefficients have been changed before the inverse transform the reconstructed shifts will differ from each other. In the de-noising case, different parts of the noise will be removed in the different shifts. By averaging the shifts the noise will be further reduced. This approach has the nice properties that it is very simple to compute and it does not introduce artifacts. That is why in our experiments we have used the averaging approach.

Beylkin's algorithm produces very smooth results which makes the images to look blurred. On the other hand, it reduces the noise a lot and makes the regions of constant intensity to look noise clean. See Fig. 3, 5.

4 Undecimated algorithm

This algorithm is based on the idea of no decimation. It applies the wavelet transform and omits both down-sampling in the forward and upsampling in the inverse transform. More precisely, it applies the transform at each point of the image and saves the detail coefficients and uses the low-frequency coefficients for the next level. The size of the coefficients array do not diminish from level to level. By using all coefficients at each level, we get very well allocated high-frequency information. From level to level there is very small step in the width of the scaling filter - instead of 8 pixels at the third level of DWT, here its width is 5 pixels. Generally, the step is not a power of 2 but a sum with 2. This property is good for noise removal because the noise is usually spread over small number of neighboring pixels. With this transform the number of pixels involved in computing a given coefficient grows slower and so the relation between the frequency and spatial information is more precise. In the ideal case, this means removal of the noise only at the places that it really exist, without affecting the neighboring pixels.

In our opinion this algorithm gives the best results in terms of visual quality − less blurring for larger noise removal.

We have tested two reconstruction schemes. The first one applies the inverse decimated wavelet transform without the up-sampling. The second one applies the traditional inverse transform at each level and then down-sample the result in order to use it for the next level. The steps for this scheme are:

1. Compute the standard inverse transform for the final level. This gives an array that has double size in each direction.

2. Decimate (down sample) the result and use it for the next level. Decimation reduces the size of the array back to the original, so we can use it for computing the next level.

(1) smallest MSE			**(2) best visual quality**		
method	**levels**	**MSE**	**method**	**levels**	**MSE**
DWT	2	83.30	DWT	5	92.75
Beylkin's	2	177.27	Beylkin's	1	227.64
shift4+mean	2	76.37	*shift4*+mean	4	66.11
undecimated	2	136.11	undecimated	4	235.00
all average	4	83.53	*all average*	4	83.53

Table 1: Representative mean square errors for the denoised images. For comparison the MSE for the noisy image (standard deviation $\sigma = 20$) used in this evaluations is 398.91. (1) Smallest MSEs from the different number of levels. The MSE is computed relative to the original image. The mean square error measures the difference between the values of the corresponding pixels from the two images. (2) Best visual quality. This is a subjective criterion, which is a compromise between the removed noise and the blurring. When a lot of noise is removed the regions with no changing intensity should look smooth. When the blurring is not a lot, the regions of high frequency data should look clear and the detail should be detectable.

Both inverse schemes produce similar results. The statistics in the tables and the example images are computed with the use of the second inverse scheme, which is less computationally expensive.

5 Averaging algorithm

A lot of computations have to be made in this algorithm. Its storage requirements are the same as in the undecimated and Beylkin's algorithm. It produces the best compromise between noise removal and blurring. The algorithm consists of the following steps:

1. Compute one level of the wavelet transform for the original image and for its shifts by one.
2. Mix the low coefficient from the four shifts according to their origin. Use the result to compute the next level at each shift. The size of the low coefficients array is equal to the image size

The idea is to approximate the low coefficients using the four shifts and thus to acquire better estimate of what the non-noisy image might have be. We can expect this to work well because the low pass coefficients represent regions of similar data in the signal and not singularities. Therefore, interpolating them will not introduce large errors [11].

The inverse transform is done in similar fashion.

1. Average the four final low coefficient images (one from each shift).
2. Use the result from the previous step, appropriately shifted, to compute the inverse DWT for each shifts at that level. Repeat these two steps for each level.

The forward transform can be summarized with the following: Do the shifts $\rightarrow$ one level forward DWT $\rightarrow$ shift back and interlace the low-pass coefficient array $\rightarrow$ go to the beginning

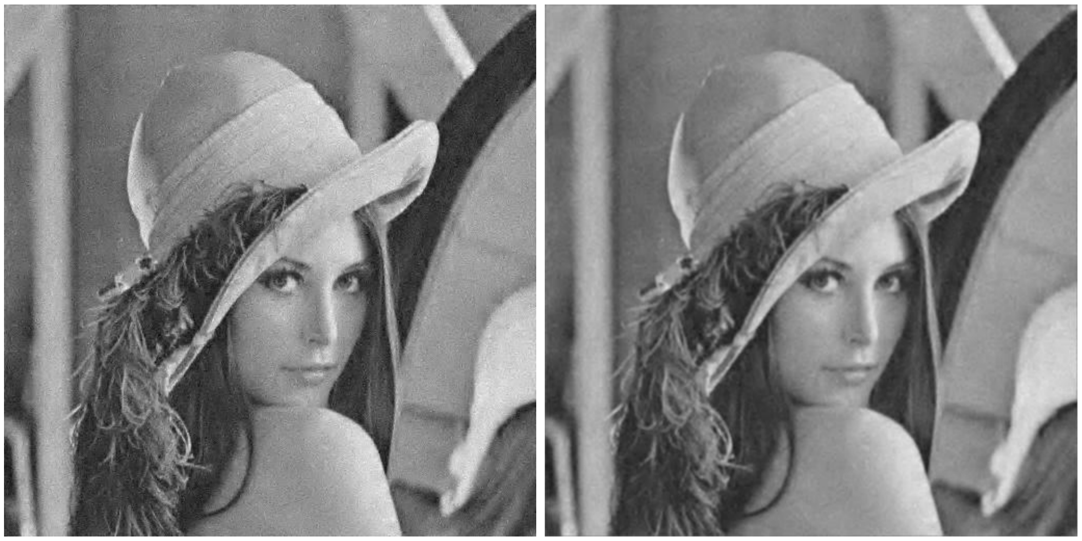
(a)  (b)

Figure 1: A result from the *shift4* algorithm with and without post filtering. (a) *shift4*, $\sigma = 20$, 4 levels. MSE=93.17. (b) the image in (a) followed bt 3×3 mean filter. MSE=66.11.

The inverse: Shift back and average $\rightarrow$ shift forward and spread the averaged $\rightarrow$ one level inverse DWT $\rightarrow$ goto the beginning

From all undecimated algorithms we have tested, this is the one that has the second smallest mean square error for every σ (noise standard deviation) value. Smaller even than the DWT when $\sigma > 10$, see Table 2. De-noised images with this algorithm are smoother than the ones produced with the undecimated algorithm. However, they are not as smooth as ones from the Beylkin's algorithm (Fig. 3). On the other hand, it removes more noise than the undecimated algorithm and less than the Beylkin's algorithm, see Fig. 5. We'll refer to this algorithm as *all average*.

6 Shifted algorithm

Undecimated algorithms produce better de-noising results than the decimated wavelet transform. Their only drawback is that they are more computationaly expensive and need larger storage space. We have tried undecimated algorithm that computes the shifts only at the first level. The algorithm applies the DWT independently over the original image and its shifts by one. (In vertical, horizontal and diagonal direction.) In this case the storage space requirements are three times bigger than the DWT, for any number of levels. The inverse transform is done also independently and at the end the four results are averaged into one. This algorithm has very straightforward implementation. We'll refer to this algorithm as the *shift4* scheme.

Unfortunately, despite of its small mean square error (MSE) values, this algorithm does leave noise in the filtered images. To obtain good visual quality we are processing the results further with 3×3 mean filter. This both keeps the MSE error small - the smallest among all algorithms and also makes the visual quality better. There is almost no blurring, see Fig. 1. The biggest advantage of this algorithm is the lower memory requirement.

Different versions of the *shift4* algorithm can be made with averaging the low frequency coefficients in the forward or inverse transform, or in both of them. This, however, will involve more computations and will increase the cost of the algorithm. The experimental results have shown only slightly better performance in this cases.

7 Wavelet thresholding for noise reduction

There are extensive studies on thresholding the wavelet coefficients for signal denoising [3], [4], [7], [8], and [9]. The so called wavelet shrinkage algorithm consists of the following steps:

1. Perform the forward wavelet transform.

2. Estimate a threshold.

3. Choose shrinkage rule and apply the threshold according to this rule.

4. Perform the inverse transform using the thresholded coefficients.

In our experiments we have used the *universal* threshold, *soft* shrinkage rule and scaled MAD (median absolute deviation) noise estimator computed from the HH coefficients at the finest wavelet transform level. For more details see [8]. The *universal* threshold is given by: $\lambda = \sqrt{2logN}$. where N is the size of the coefficient arrays. We have computed N on the level dependent basis. For the Beylkin's and *shift4* algorithms the value of N changes for each level. For the undecimated and *all average* algorithm it is a constant.

The shrinkage rule define how we apply the threshold. There two main approaches. The so called hard thresholding deletes all coefficientss that are smaller than the threshold λ and keeps the others unchanged. The hard thresholding is defined with: $\delta_\lambda(w) = sgn(w)(|w|)_+$, where λ is the threshold and the plus sign indicates only the coefficients that are above the threshold are considered.

The other one - soft thresholding also deletes the coefficients under the threshold, but scales the ones that are left. There are different ways of scaling. The general soft shrinkage rule is defined by: $\delta_\lambda(w) = sgn(w)(|w| - \lambda)_+$.

The scaled MAD noise estimator is computed by: $MAD = median(|w_i|)/0.6745$, where w_i are the HH coefficients form the finest decomposition level.

We have made experiments with Symmlet, Coiflet and Daubechies wavelets. The results were similar for the different wavelet basis. The experimental results that are shown in the figures and tables are for the Daubeshies bases with length 4.

8 Results

For our experiments we have used images corrupted with additive Gaussian noise with $N(0, \sigma^2)$. For numerical evaluation we have used the mean square error relative to the original image. Figure 3 shows the results from denoising the "Lena" image. The results for other images have followed the same tendency.

Figure 4 shows an enlarged detail from the images, so the blurring effect can be evaluated better.

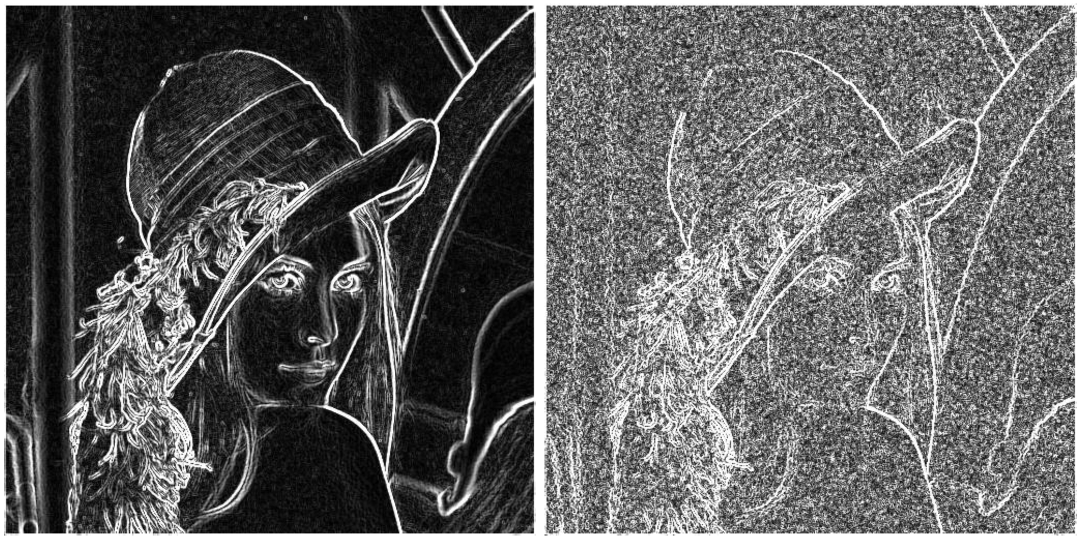
(a)  (b)

Figure 2: Edge detection results from the original image and the noise corrupted image with $\sigma = 20$. (a) Original image. (b) Noisy image.

Figure 2 shows the edge detection applied to the original images and to the noise corrupted image with $\sigma = 20$. Edge detection gives better idea of how much noise is removed and how well the edges are preserved. In the image processing algorithms, it is often the next step after the de-nosing step. Figure 5 shows the edge magnitude resulting from the Sobel edge detection operator applied over the same de-noised images.

9 Summary

We have designed three different undecimated wavelet transform schemes and tested their performance for image denoising. Comparision with the performance of the standard discrete wavelet transform was made. Results show that the undecimated transforms produce better results, although their mean square error is oftenly higher than the DWT error. Our test set uses white additive non-correlated noise. Those algorithms might not show the same behavior for different kind of noise.

We have made experiments with different wavelet bases observing no significant change in the final results. We can conclude that for our experiment set the choice of the wavelet basis is not crucial.

Throughout this work we have noticed the big importance of the threshold value. Small changes in it mresults in big changes in the de-noised image. The *universal* threshold makes general assumption about the noise. This means that more sophisticated choice of threshold might improve the performance a lot.

Finally, we'll note that the quality criterion is very relative. Depending on the type of images and the scale of the objects in them one may prefer different algorithms. For example, in astronomical images, the quality criteria depend on different features, compared to face images. De-noising algorithms might be better if they involve not only the noise, but also the image spatial characteristics.

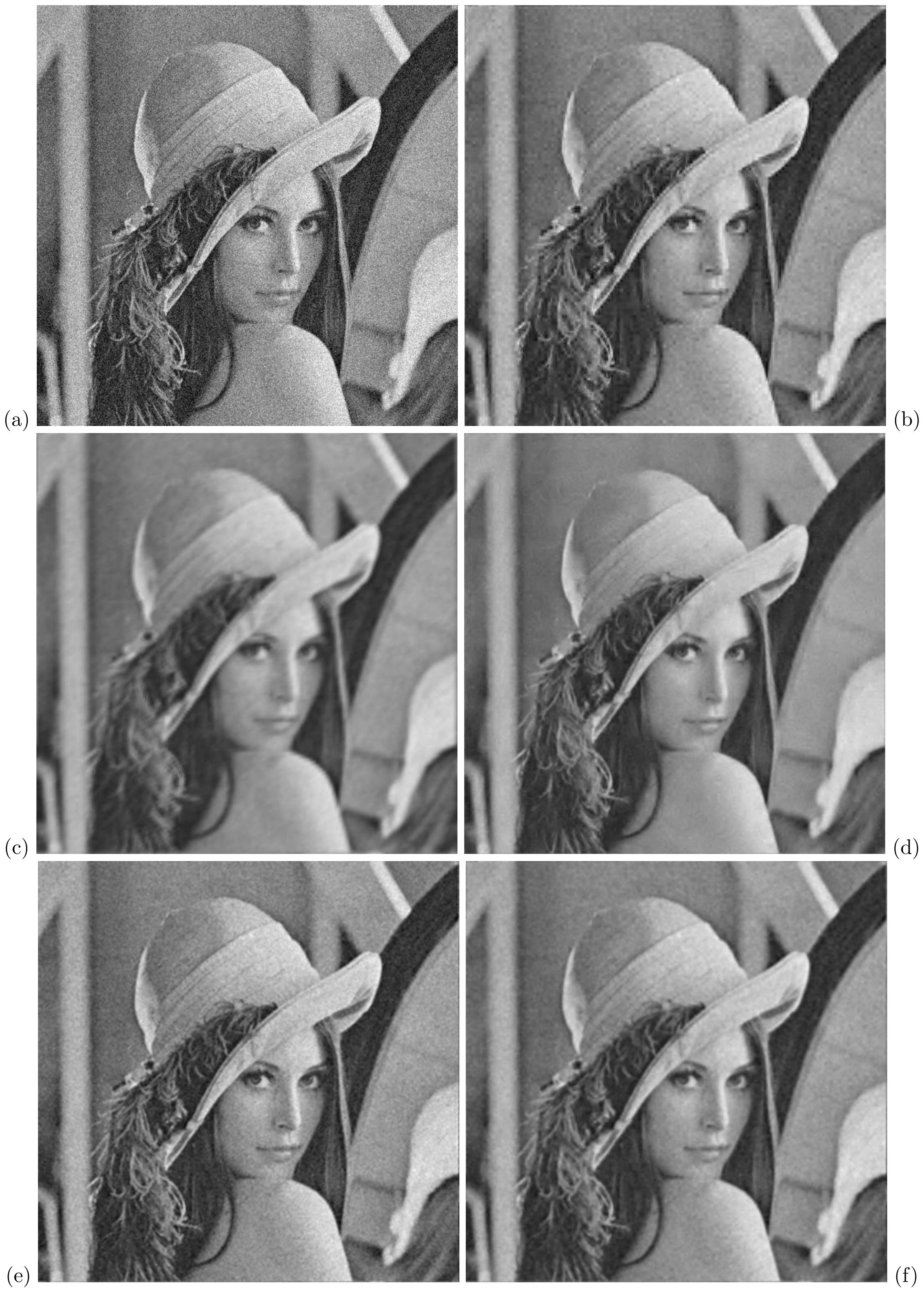


Figure 3: Comparison among different techniques. $\sigma = 20$, Daubechies wavelet with length 4, *Universal* threshold and soft shrinkage. (a) Original image. (b) DWT. (c) Beylkin's. (d) *Shift4* + mean filter. (e) Undecimated. (f) *all average*

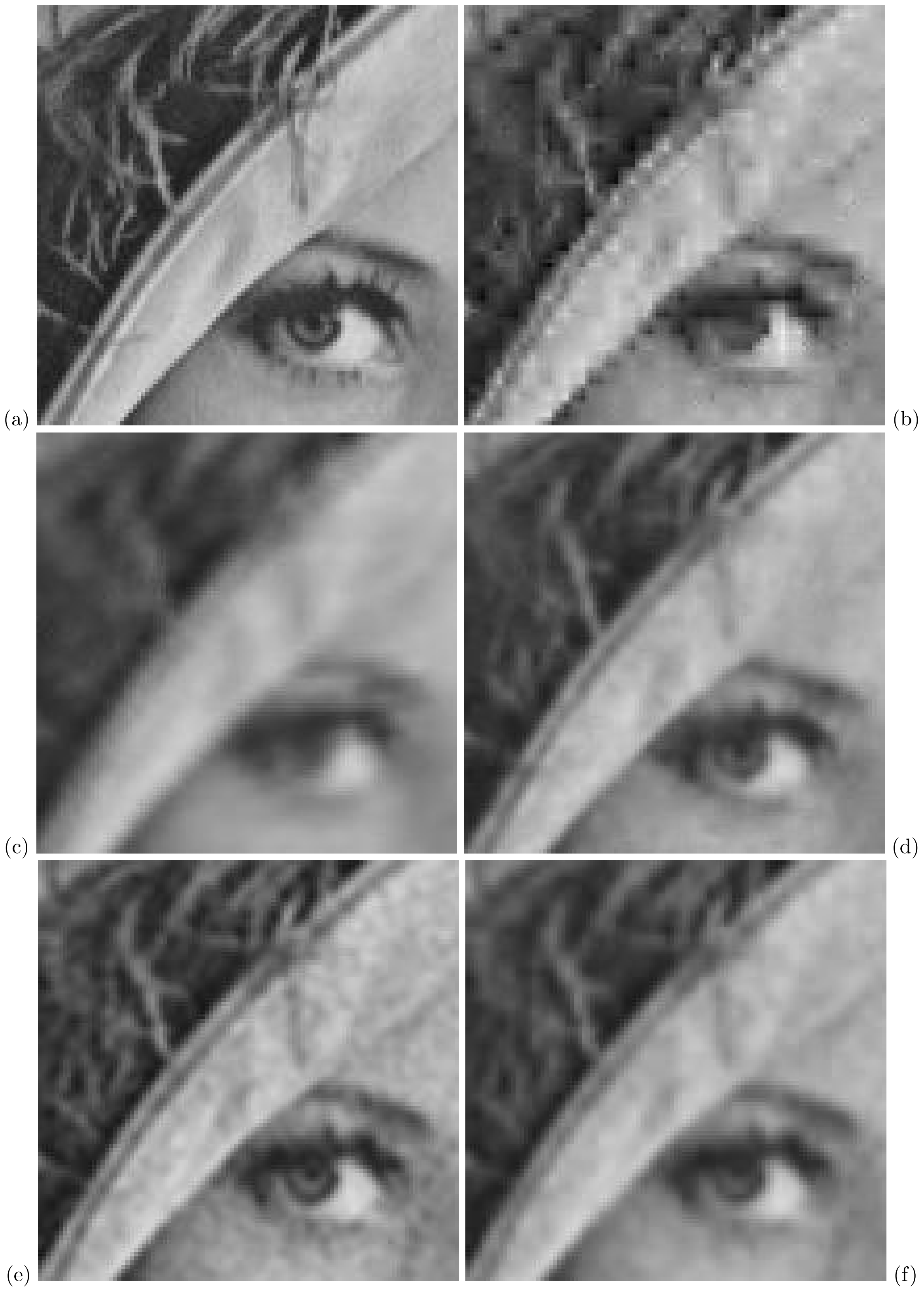


Figure 4: The same test settings as in Fig. 3. Here a detail from the image is shown for blur comparision. (a) Original image. (b) DWT. (c) Beylkin's. (d) *Shift4* + mean filter. (e) Undecimated. (f) *all average*.

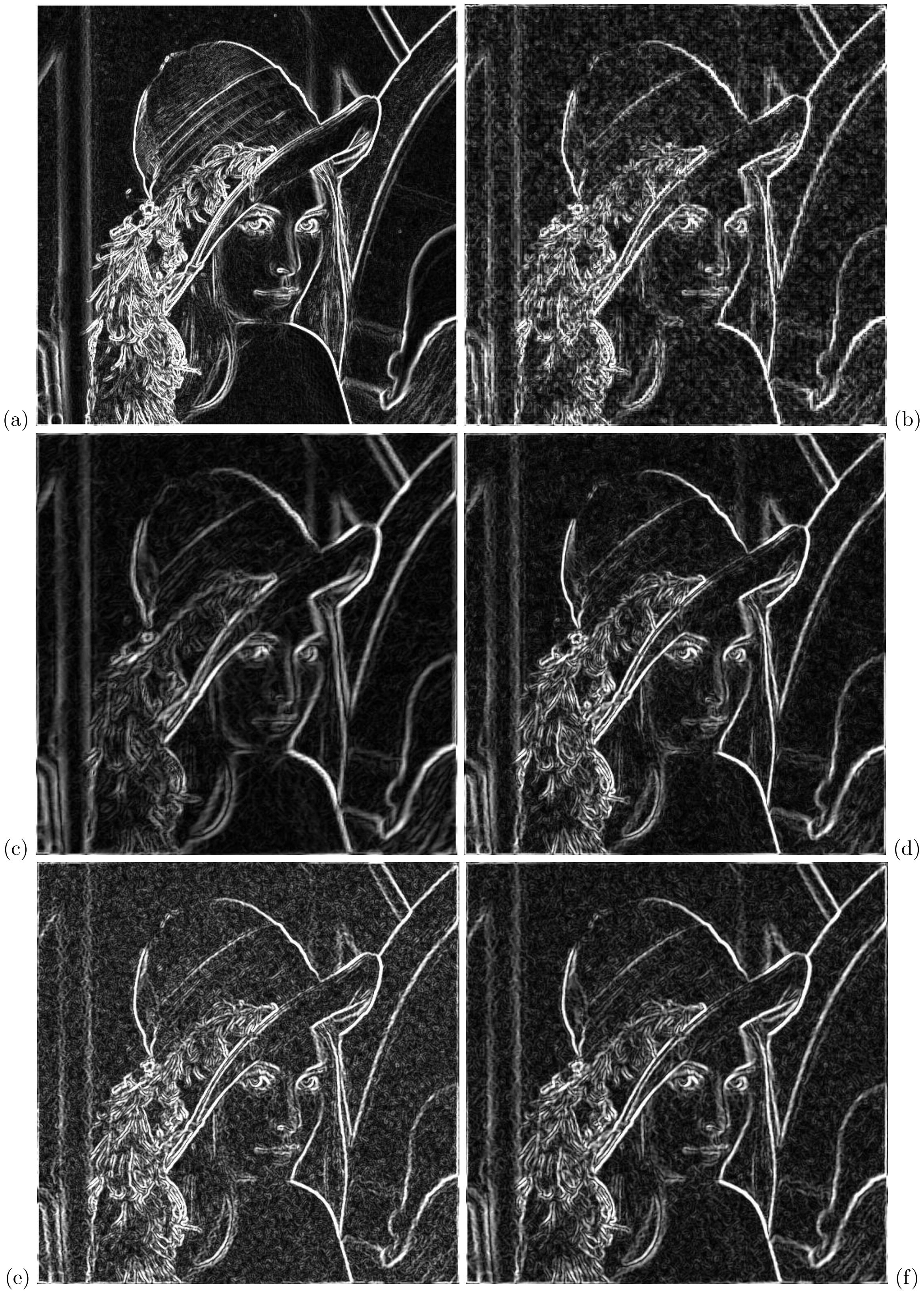


Figure 5: The same test settings as in Fig. 3 followed by edge detection with Sobel operator. (a) Original, noise free, image. (b) DWT. (c) Beylkin's. (d) *Shift4* + mean filter. (e) Undecimated. (f) *all average*.

method	**levels**	$\sigma = 10$	$\sigma = 20$	$\sigma = 30$
DWT	3	65.90	85.80	153.04
DWT	4	72.30	90.70	180.94
DWT	5	74.62	92.75	197.35
Beylkin's	1	202.63	277.64	270.08
Beylkin's	2	169.06	177.27	188.30
Beylkin's	3	229.36	239.77	247.45
shift4+mean	3	46.07	64.53	88.25
shift4+mean	4	46.07	66.11	89.92
shift4+mean	5	46.43	66.84	91.17
undecimated	2	95.13	136.40	186.83
undecimated	4	201.15	235.00	269.23
undecimated	6	328.85	360.61	394.70
all average	4	69.10	83.53	107.99
all average	5	68.09	82.37	104.95
all average	6	108.19	122.51	143.32

Table 2: The **MSE** values for different algorithms and for different σ values. We want to remind that the small MSE values did not always correspond to good visual quality. (For example, the undecimated algorithm produced one of the best results in terms of visual quality.)

UCRL-ID-150931. This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

References

[1] G. Beylkin. *On the representation of operators in bases of compactly supported wavelets.* SIAM J. Numer. Anal., 29 (1992), 1716-1740

[2] I. Daubechies. *Ten lectures on Wavelets.* SIAM, Philadelphia, PA, 1992.

[3] D.Donoho. *De-noising by soft thresholding.* IEEE Trans. on Information Theory, vol. 38(2), pp. 613–627, 1995.

[4] D. Donoho and I. Johnstone. *Ideal spatial adaptation by wavelet shrinkage.* Biometrika, vol. 81(3), pp. 425-455, 1994

[5] H. Guo. *Theory and Applications of the Shift-Invariant, Time-Varying and Undecimated Wavelet Transforms.* Master's Thesis, Dept. ECE, Rice University, Houston, TX, 1995

[6] Holschneider, M., Kronland-Martinet, R., Morlet, J., and Tchamitchian, Ph.: *A real-time algorithm for signal analysis with the help of the wavelet transform*, in Wavelets, Time-Frequency Methods and Phase Space, J.M. Combes, A. Grossmann, and Ph. Tchamitchian (eds.), Springer-Verlag, Berlin, pp. 286–297, 1987.

[7] M. Jansen. *Noise reduction by wavelet thresholding.* Lecture Notes in Statistics, vol.161, Springer-Verlag, 2001.

[8] I. Fodor and C. Kamath. *On denoising images using wavelet-based statistical techniques.* Lawrence Livermore National Laboratory technical report, UCRL JC-142357

[9] M. Lang, H. Guo, J.E. Odegard, and C.S. Burrus, *Nonlinear processing of a shift invariant DWT for noise reduction*, SPIE, Mathematical Imaging: Wavelet Applications for Dual Use, April 1995.

[10] S. Mallat. *Zero-crossings of a wavelet transform.* IEEE Trans. Inform. Theory, 37(4), July 1991.

[11] F.G. Meyer, A. Averbuch, and R.R. Coifman, *Motion compensation of wavelet coefficients for very low bit rate video coding*, Intl. Conf. on Image Processing, ICIP'97, Santa Barbara, Oct. 1997.

[12] G. Nason and B. Silverman. *The stationary wavelet transform and some statistical applications*, in Wavelets and Statistics (Antoniadis and Oppenheim eds.), 281–299, 1995.

[13] M. J. Shensa. *The discrete wavelet transform: wedding the à trous and Mallat algorithms.* IEEE Trans. Inform. Theory, 40:2464-2482, 1992.

[14] M. V. Wickerhauser. *Adapted wavelet analysis from theory to software.* AK Peters, 1994.
http://www.math.wustl.edu/ victor/awaftts/